cite

SEARCH RESULTS [PDF Full-Text (1204 KB)] PREVIOUS DOWNLOAD CITATION

The virtual wind tunnel

- Bryson, S. Levit, C.
NASA Ames Res. Center, Moffett Field, CA, USA

This paper appears in: IEEE Computer Graphics and Applications
On page(s): 25 - 34
July 1992
Volume: 12 Issue: 4
ISSN: 0272-1716
References Cited: 12
CODEN: ICGADZ
INSPEC Accession Number: 4260795

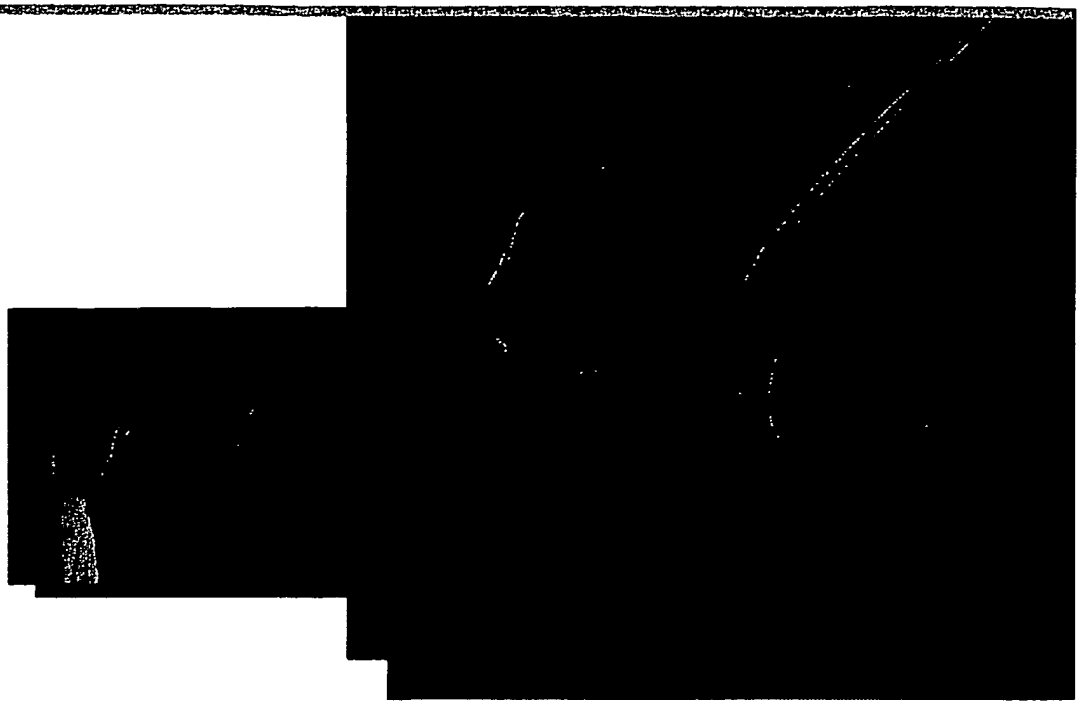
Abstract:
The design and implementation of a virtual environment linked to a graphics w for the visualization of complex fluid flows are described. The system user wea head-tracked display, which effectively displays 3-D information, and an instru glove to intuitively position flow-visualization tools. The visualization structure interfaces in the virtual environment and the implementation hardware and so described. The performance of the virtual wind tunnel is reviewed using the flo tapered cylinder as an example. Performance issues and future directions for t are discussed.

Index Terms:
3D information virtual wind tunnel virtual environment graphics workstation v complex fluid flows stereo head-tracked display instrumented glove tapered cy CAD computer graphics flow visualisation

Documents that cite this document
Select link to view other documents in the database that cite this one.

SEARCH RESULTS [PDF Full-Text (1204 KB)] PREVIOUS DOWNLOAD CITATION

Home | Log-out | Journals | Conference Proceedings | Standards | Search by Author | Basic Search | Advan
Join IEEE | Web Account | New this week | OPAC Linking Information | Your Feedback | Technical Support | F
No Robots Please | Release Notes | IEEE Online Publications | Help | FAQ| Terms



The Virtual Wind Tunnel

Steve Bryson
Computer Sciences/NASA Ames Research Center

Creon Levit
NASA Ames Research Center

A virtual environment for exploring numerically generated 3D unsteady flow fields employs a boom-mounted CRT headset for viewing and a glove controller for injecting tracers.

In the velocity vector fields that result from numerical flow simulations, complicated geometrical and topological situations abound, making visualization of the 3D flow fields difficult. Multiple vortices, recirculation bubbles, and chaotic flows within vortex breakdown have all been observed in computer simulations of *steady* 3D fluid flows. Some researchers have worked on the visualization of numerically computed unsteady fluid flows,[1-3] but the greater complexity involved demands new techniques for effective visualization.

A fruitful source of new methods of numerical flow visualization is the classical physical techniques—those used to visualize real flows in real wind (or water) tunnels.[4] Smoke injection, dye advection, time exposure photographs, and the placement of tufts or streamers into the flow are examples of these classical techniques. Additional physical flow-visualization techniques include Schlieren interferometry, laser sheet illumination, stroboscopic illumination, and injection of tracers sensitive to fluid properties such as temperature. Computational analogs of these techniques are all feasible using modern high-performance graphics workstations or distributed computing. These computational analogs of classical wind tunnel techniques might prove useful in visualizing other vector fields as well.

0272-1716/92/0700-0025$03.00 © 1992 IEEE

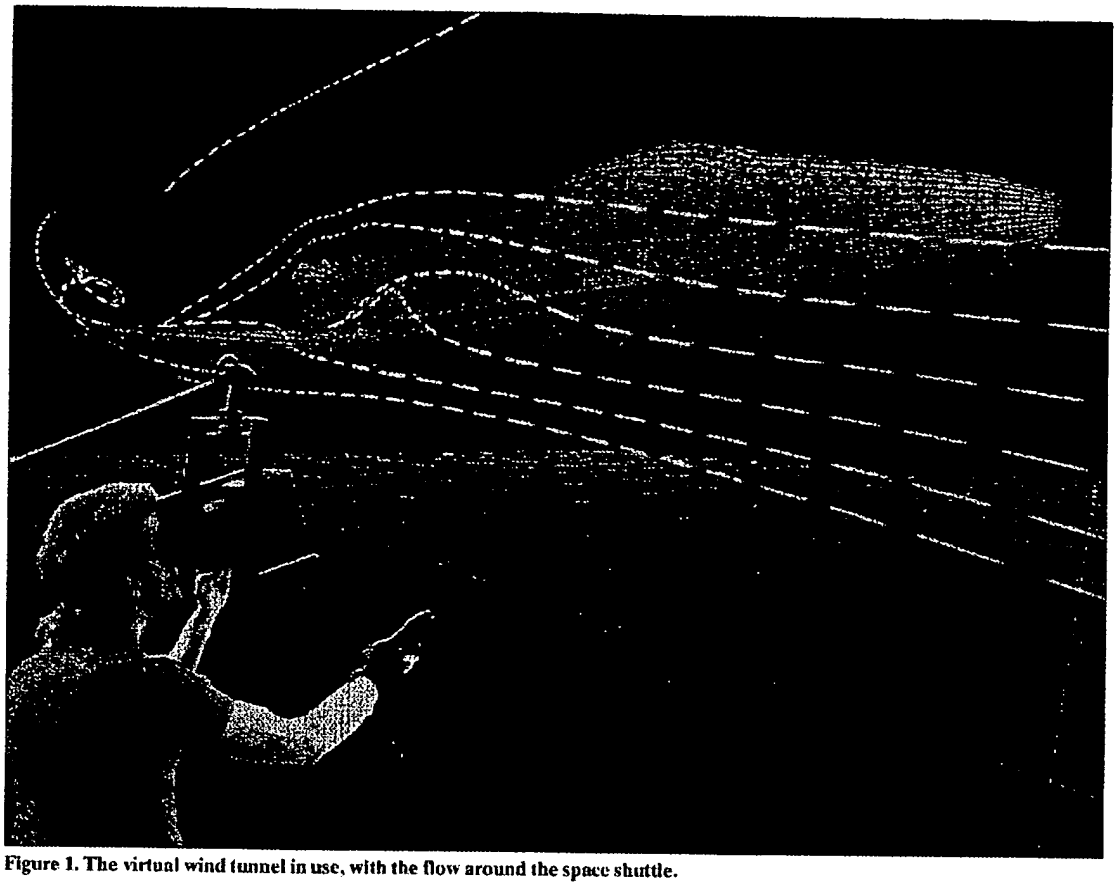


Figure 1. The virtual wind tunnel in use, with the flow around the space shuttle.

The computer system requirements for unsteady flow visualization are substantial. They include fast computation, quickly rendered high-resolution graphics, and massive data storage and retrieval capabilities. The amount of solution data produced by a single 3D unsteady fluid dynamics calculation can be quite large—several thousand megabytes or more. High-performance graphics workstations have now reached the level where real-time interactive exploration of some 3D unsteady-flow-field solutions is possible.

In this article, we describe the design and implementation of a virtual environment linked to a graphics workstation for the visualization of complex fluid flows. The user wears a stereo head-tracked display, which effectively displays 3D information, and an instrumented glove to intuitively position flow-visualization tools. As Figure 1 shows, the idea is to create for the user the illusion that he or she is actually in the flow manipulating the visualization tools. Unlike someone in a real flow field, however, the user's presence in no way disturbs the flow. Thus, sensitive areas of flow, such as boundary layers and chaotic regions, can be investigated easily. Further, since the flow is precomputed, it can be investigated at any length scale and with control over time. For detailed examination, the user can speed up, slow down, run backward, or stop completely the time evolution of the flow.

After we describe the visualization structures and their interfaces in our virtual environment, we describe the implementation hardware and software. Then we discuss and review the performance of the implementation of the virtual wind tunnel, using flow past a tapered cylinder[5] as an example.

In this article, when we refer to a flow field, we mean a numerical solution to a 3D computational fluid dynamics simulation, and in particular the time-dependent velocity-vector-field part of the solution. The flow fields we consider are precomputed solutions of the time-accurate Navier-Stokes equations of fluid motion. These unsteady flow fields are represented as a collection of successive 3D velocity vector fields. We consider each of these velocity vector fields as a time step.

When we refer to virtual environments, we mean the integration of input and display devices designed to give users the illusion of being immersed in an interactive computer-gener-

ated environment. Stereo display of the computer-generated scene creates the illusion of depth. The scene is rendered from a point of view that tracks the user's head. Input devices give users the experience of directly manipulating objects in the computer-generated environment.

Visualization tools and interfaces

Many techniques assist flow visualization, such as isosurfaces, color-mapped cutting planes, streamlines, and rendering of vectors. The tools we consider here for visualizing unsteady velocity vector fields are inspired by classical techniques used in real wind and water tunnels. Currently, the visualization methods or tools we have implemented are tufts, streaklines, particle paths, and streamlines. Our choice was motivated by ease of use as well as performance considerations.

Definitions

The simplest visualization technique we use is the placing of *tufts*, or small vanes, into the flow field.[6] These tufts, each anchored at a particular location in space, allow direct visualization of the changing velocity vector at that location.

A *streakline* is formally defined as the locus of infinitesimal fluid elements that have previously passed through a given fixed point in space.[7] Informally, a streakline is the evolving curve you obtain if you continuously inject from a fixed location a stream of tracer particles into the flow. In water tunnels, streaklines are usually visualized by generating hydrogen bubbles rapidly, one after another, at a single place with an electrolytic wire. Thus, we call the seed point for a streakline a *bubbler*.

A *particle path* is formally defined as the locus of points occupied over time by a given single, infinitesimal fluid element.[7] Informally, a particle path is the curve you obtain if you take a "time exposure photograph" of the motion of a single small particle injected into the flow.

A *streamline* is formally defined as the integral curve of the instantaneous velocity vector field that passes through a given point in space at a given time.[7] Despite their transitory and global nature, streamlines can be visualized in water tunnels by having many tracer particles in the flow and taking a brief time exposure photograph. The many short, straight particle paths so obtained can be connected to form streamlines.

For steady flows, streaklines, streamlines, and particle paths all coincide. However, for unsteady flows, they are distinct. This has led to some confusing terminology in the numerical flow-visualization literature, since most previous work has been done on steady flows.

Implementation

Except for tufts, the numerical flow-visualization techniques introduced in the previous section involve injecting virtual massless point particles into the flow and integrating their trajectories. The distinction between these techniques lies in how the particle positions are integrated relative to the time evolution of the field and how they are displayed. We call the point of injection for a particular tool the seed point for that tool. In this section, we outline the various computations used for each tool.

Streaklines consist of point particles, or bubbles, all of whose positions change at every time step. The seed point or bubbler continuously emits new bubbles at a specified rate, usually once per time step. The position of a bubbler itself is considered static.

A bubble is a particle with a position (x, y, z). At each time step, the system computes the bubble's new position by integrating the velocity vector field over the current time step and updating the bubble's position. This position is displayed at the next time step, during which the bubble is again moved.

Each bubble is treated independently. Typically, a bubbler emits a certain number of bubbles, after which its oldest bubbles are recycled back to the seed point in the order emitted. For our purposes, a streakline, at any time step, is the current positions of all bubbles emitted by a particular bubbler (see Figure 2).

A particle path is akin to a time exposure of the trajectory of a particle released at a particular seed point at a particular time. In an unsteady flow, the particle path from a particular seed point will vary, depending on when the particle is emitted. The system computes the particle path by integrating the position of the seed point through the unsteady velocity field over each successive time step and storing its successive positions into an array. The trajectory is not displayed until the integration is

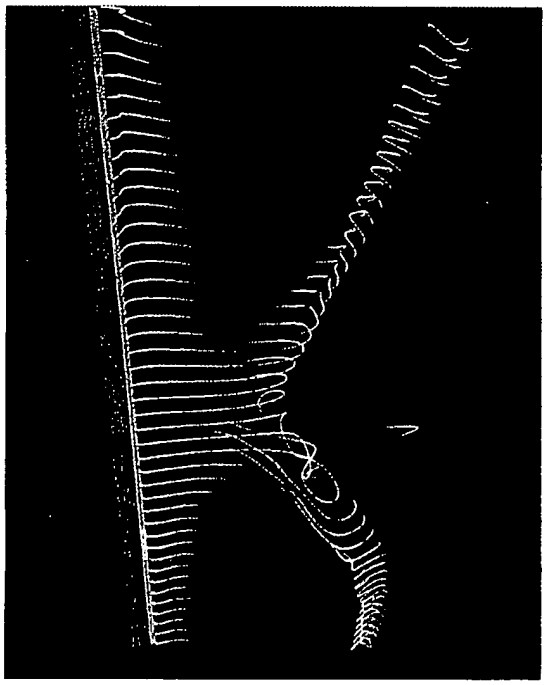


Figure 2. Streaklines of the flow around the tapered cylinder rendered as smoke.

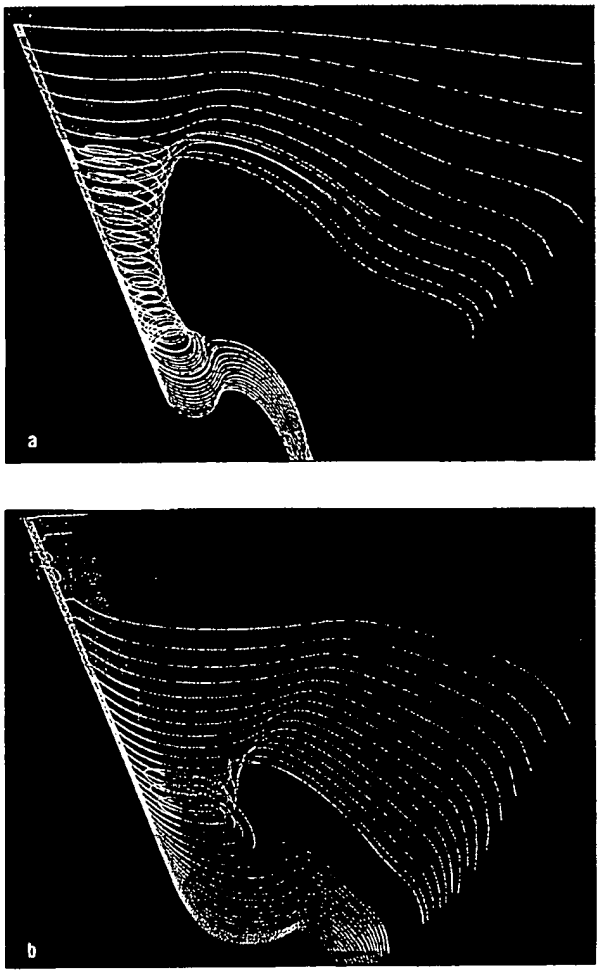


Figure 3. Streamlines of the flow around the tapered cylinder at two successive moments of time.

complete, at which time the trajectory is displayed in full. The complete path of a particle emitted at the current time step is computed and displayed anew at each time step. It is displayed as a curve extending from the seed point, representing the path that the particle currently at the seed point will take.

A streamline is computed like a particle path, except that the integration is carried to completion using only the single vector field corresponding to the current time step. The streamline is nonphysical in the sense that no fluid element actually follows its trajectory. Nevertheless, streamlines, being integral curves, do give insight into the global structure of the velocity vector field, as Figure 3 shows.

Tufts are simply very short streamlines. The system can usefully approximate them by drawing the velocity vectors themselves.

Interface

The use of the tools we have described involves the placement of their seed points and the viewing of the graphic objects that result. There are several approaches to the placement of seed points. One approach is to have the computer automatically place them based on analysis of the flow field. For example, it can place seed points near critical points of the vector-field topology[8] or near local maxima of interesting scalars such as helicity.[9] A second approach is to have the user specify in advance the positions of the seed points, usually as textual arguments to some command. Although this may sound primitive, it is often most useful. A third approach is simply to give the user flexible, rapid, interactive control over the placement of seed points. Our virtual environment supports the last two approaches.

Besides rapid placement of seed points, our virtual environment allows for quick, intuitive repositioning and deletion of existing seed points. Users can group multiple seed points together into *rakes*, which can be repositioned and deleted. This feature is important because the flow visualization may involve large numbers of each of the flow visualization tools. For example, our experience shows that about 40 bubblers emitting 100 bubbles each are a minimum requirement for effectively visualizing interesting features of a flow using streaklines. Further, moving these bubblers to another location may reveal previously unseen flow features. These considerations apply to particle paths, streamlines, and tufts as well.

Various properties of the rakes, such as the type of visualization associated with the rake, the number of seed points in the rake, or the length of the streamlines, are controlled via a screen/text interface outside the virtual environment. Using the boom display system (described later), we find it fairly convenient to enter and exit the virtual environment to control various aspects of the visualization tools.

The user manipulates a rake with his or her hand using one of three grab points: the middle and two ends. If the user's hand position coincides with the rake's center, forming a fist causes the entire rake to move rigidly with the hand's position. This permits repositioning of the entire rake. If the user's hand position coincides with one end of the rake, forming a fist causes that end to move while the other end stays in place. In this way, the user can change the rake's orientation at will. This method of controlling rake orientation offers more control and versatility than rotation of the rake about its center. The system draws a wireframe sphere when the user's hand position coincides with one of these three grab points, even if no gesture is formed. The sphere shows where the rake will be grabbed if the user forms a fist (see Figure 4).

The user selects a rake as the current rake by putting his or her hand near the rake's center grab point. Using hand gestures such as pointing with various fingers, the user can copy or delete the rake, or perform other operations on it. The user can define

several rakes, each with its own properties, to explore different features of a flow in different locations simultaneously.

Since fluid dynamic phenomena occur over a large range of scales (several orders of magnitude in space), navigation through the virtual environment is more difficult than, say, architectural walkthroughs. Thus, in addition to "standard" viewing capabilities sensitive to head position and head orientation, our virtual environment lets the user rapidly change his or her scale, or the scale of the environment. Relative to the environment, a user can shrink to become completely surrounded by some small vortex or grow so that the entire flow field fits in one hand.

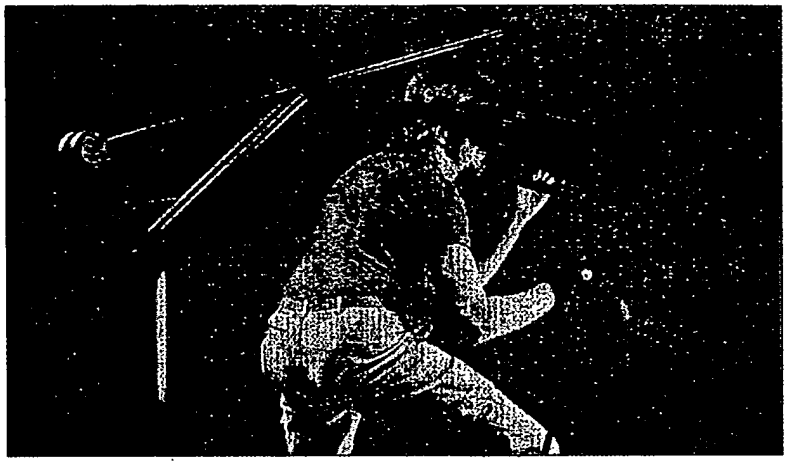


Figure 5. Boom and glove hardware interface to the virtual wind tunnel.

The flow-visualization tools produce 3D structures that may wind through space in complex and even chaotic ways. Cues illuminating the 3D geometry help the user get a good mental picture of the flow field. Without these cues, ambiguity can result, leading to poor perception of the flow field. In our virtual environment we use a combination of several techniques to disambiguate 3D structures. These cues include real-time 3D rendering in response to head position and orientation to provide perspective, motion parallax, and vestibular cues, and z-buffering to enhance image realism through selective obscuration. Wide field-of-view optics provide input to the peripheral visual field and give a realistic, compelling optical flow. Stereo display provides binocular parallax for greater perception of 3D structure and further widens the field of view. Finally, feedback as to the position of the hand and animation of the streamlines and particle paths themselves provide additional 3D cues.

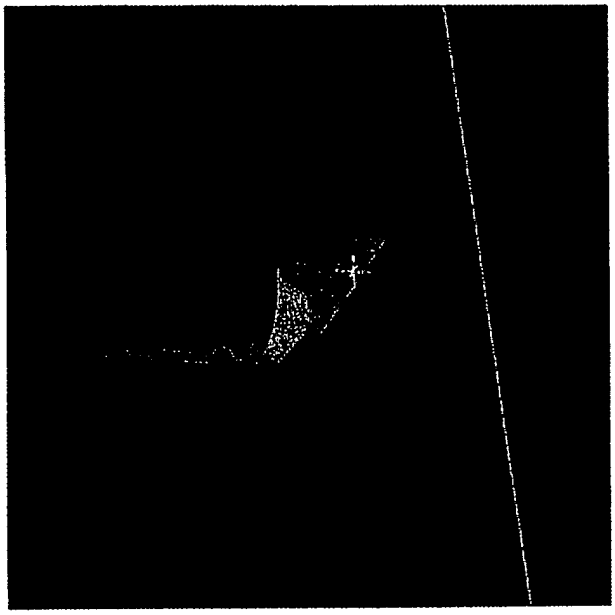


Figure 4. User interaction with rakes. There are two rakes in the figure: one very long vertical rake emitting bubbles that indicate vortex structure, and a short horizontal rake with short streamlines. The cross-shaped cursor shows the user's hand position as sensed by the VPL Dataglove, and the dark sphere on the smaller rake indicates that the middle grab point of that rake will be "picked up" if the user makes a fist gesture.

Moving rakes in an unsteady flow raises the issue of determining what dynamic behavior (for example, streamlines) is due to the motion of the rake and what is due to the changing of the vector field over time. We address this issue by allowing the user to control the flow of time in the virtual wind tunnel. Time can be stopped, stepped, run backward, or run at various speeds. Currently, the user controls time flow with keyboard commands.

Hardware

Perhaps the most interesting hardware component of our virtual environment is the boom-mounted display, shown in Figure 5. This boom supports two small CRTs on a counterweighted yoke attached through six joints to a base. It is manufactured by Fake Space Labs of Menlo Park, Calif., and fashioned after the prototype

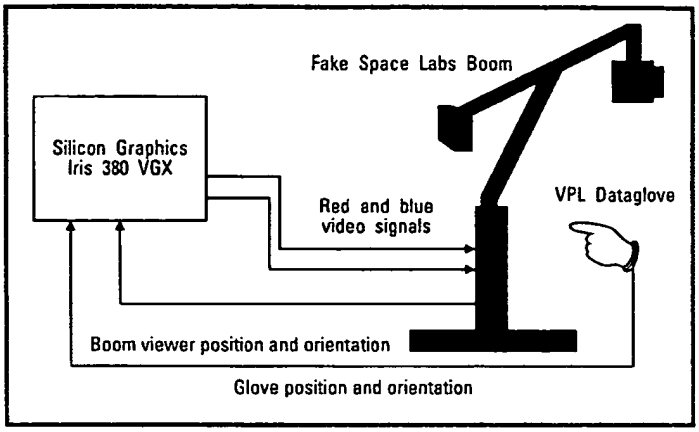


Figure 6. The hardware configuration of the virtual wind tunnel system.

developed earlier by Sterling Software at the Virtual Interactive Environment Workstation (VIEW) lab at NASA Ames Research Center.[10]

The boom is an alternative to the popular head-mounted LCD display systems pioneered at the View lab.[11] While head-mounted displays have the advantage of unencumbered movement, weight and size constraints in head-mounted systems limit these displays to low-resolution LCDs. The virtual wind tunnel requires higher image quality and resolution than are currently available in LCD displays. The main advantage of the boom is that real CRTs can be used for display despite their mass, since the user bears none of their weight. CRTs have much better brightness, contrast, and resolution than standard LCDs.

The CRTs are mounted on the "head" of the boom, along with the wide-field optics and a multifunction handle. This handle facilitates the user's movement of the boom and has two buttons and a pressure sensor (not used in the current implementation). The boom's gimbals and joints provide six degrees of freedom of motion in a smooth and force-free manner. Within a very wide range, the user can continuously change the 3D position and orientation of the boom's head. The position and orientation information is based on the current state of the six joint angles. These angles are sensed by optical encoders at the joints, which deliver a known value per degree of turn. These values are fed into a microprocessor in the base of the boom, which formats the information and transmits it via a serial (RS-232) port. No magnetic-field emitters or sensors are used, so the boom information is precise, repeatable, and insensitive to the electromagnetic environment.

Originally, the CRT monitors on the boom are monochrome. The boom accepts two RS-170 video signals, one for each eye. The monitors have been upgraded to 1,024 × 1,024-pixel displays with two color channels (red and green components).

The boom's motion is relatively effortless and completely smooth. First-time users are universally surprised that a structure of this size moves so easily. Admittedly, compared with head-mounted displays, the footprint of the device is large and the freedom of motion restricted. But when used from a sitting position in a wheeled office chair, it provides ample freedom of motion, as it does with the user standing. It has no straps, puts no weight on the head, and is easy to disengage and hand to another user.

In addition to the user's head position and orientation, the user's hand position, orientation, and finger joint angles are sensed using a VPL Dataglove, Model II, which incorporates a Polhemus 3Space tracker. The Polhemus tracker gives the absolute position and orientation of the glove relative to a source by sensing multiplexed orthogonal electromagnetic fields. With this data, the host computer plots the position and orientation of the user's hand. The Dataglove measures the degree of bend of the knuckle and middle joints of the user's fingers and thumb with specially treated optical fibers. The system interprets as gestures these finger joint angles using simple pattern-matching techniques.

The glove requires recalibration for each user, and the Polhemus tracker on the glove is, unfortunately, sensitive to the room's electromagnetic environment. Nevertheless, this part of the system works reliably and satisfactorily, once calibrated for a user's hand and a room's magnetic peculiarities.

The keyboard and mouse also serve as input devices to the virtual environment. The user can easily swing the boom away and refocus on the normal computer screen. The user, whether standing or sitting, can then return to typing and interacting with the computer in the usual way. For small amounts of typing and for controlling the mouse, the user need not remove the glove, since it is quite thin and flexible.

A Silicon Graphics Iris 380 VGX system provides the computation and rendering for our virtual environment. This multiprocessor system has eight 33-MHz RISC processors (MIPS R3000 CPUs with R3010 floating-point chips). The machine's performance is rated at approximately 200 million instructions per second (200 VAX MIPS) and 37 million floating-point operations per second (37 64-bit Linpack Mflops). Our system currently has 256 Mbytes of memory.

The Iris 380 VGX has parallel hardware rendering pipelines. The rated graphics performance of our system is around 800,000 small 3D triangles transformed, clipped, projected, lighted, shaded, and displayed per second. The system has over 200 bits per pixel of frame-buffer memory. We use only 48 bits per pixel: two buffers each of eight bits of red and eight bits of blue (double buffering), and 24 bits of z-buffer.

For stereo display on the boom, the system renders the left-eye image using only shades of pure red (of which 256 are

available) and the right-eye image using only shades of pure blue (of which 256 are available). When it draws the blue (second, right-eye) image, it uses a "writemask" that protects the bits of the red image. The z-buffer bit planes are cleared between the drawing of the left- and right-eye images, but the color (red) bit planes are not cleared. Thus, the result is separately z-buffered left- and right-eye images, in red and blue respectively, on the screen at the same time with the appropriate mixture of red and blue where the images overlap.

The 1,024 × 1,280-pixel RGB video output of the Silicon Graphics Iris 380 VGX is converted into RS-170 component video in real time using a scan converter. While the VGX can put out RS-170 video directly if we set a software switch, we have found that scan converting the higher resolution 1,024 × 1,280-pixel image using dedicated hardware provides spatial and temporal antialiasing, and consequently noticeably better image quality when viewing with the boom. The red RS-170 component is fed into the left eye of the boom, and the blue RS-170 component into the right eye. Since the boom CRTs are monochrome, we see correctly matched (stereo) images.

Figure 6 shows the system configuration for virtual-reality mode.

Software architecture

The graphics are rendered in stereo from a point of view determined by the boom. The interface to the flow-visualization tools is based on the glove position and gesture. Another interface allows movement of the flow data relative to the user.

The position and orientation of the CRTs on the boom head are determined by optical encoders mounted on the six boom joints. The output of each encoder is linearly related to its respective joint angle. The host computer system reads these six joint angles and converts them into a standard 4 × 4 position and orientation matrix. This conversion is the result of six successive translations and rotations. The rotations are rotations about the local axis of the corresponding joint by the angle read at that joint, and the translations are by the distances between joints. In the position and orientation matrix, the position is measured in meters from the boom's base.

The graphics are rendered in stereo from the point of view of the boom's viewers by inverting the boom's position and orientation matrix, translating to the left or right by half the distance between the eyes (depending on which eye's view is being drawn), and multiplying by a precomputed perspective matrix. The resulting matrix is put on the graphics transformation stack before the rendering of graphics for each respective eye. Thus, the entire view must be rendered twice.

The alignment of the resulting images in the viewer must be correct to obtain a proper stereo effect. This is accomplished by defining separate viewports for rendering the graphics for each eye, with the horizontal position of each viewport controlling the alignment.

Before rendering the graphics data, the system concatenates another transformation embodying a rotation and translation onto the transformation stack. This allows the data to be in an arbitrary position and orientation with respect to the coordinate system of the boom. This transformation can be determined in a variety of ways, such as manipulation with the Dataglove with a fist gesture ("grabbing the data"). In particular, the user can manipulate the entire graphics data via the Dataglove. This transformation is called the *data coordinate transformation.*

The Dataglove's primary use is in the placement of rakes of seed points for the various visualization tools. The interface is gesturally based.

The Dataglove's primary use is in the placement of rakes of seed points for the various visualization tools. The interface is gesturally based. As described earlier, when the glove position is matched with a rake's position and the glove is in a fist, the rake is "picked up" and follows the position of the glove until the fist gesture is released. Throughout this time, the system recomputes and renders the graphic representation for this rake, allowing the user to observe the tool as it moves. When the user performs another gesture, other actions may occur, such as a new rake being placed at the glove's current position.

The interface actually spans three coordinate systems: the glove position is in boom coordinates, the graphics data are in data coordinates, and the streamlines are in computational coordinates. To transform from boom coordinates to data coordinates, the system multiplies the glove position vector by the inverse of the data coordinate transformation. The resulting vector is in data (x, y, z) coordinates.

The seed point positions are in computational (ξ, η, ζ) coordinates, which are defined on a discrete curvilinear computational grid. The values (x, y, z) in data coordinates for the grid vertices are stored in a 3D array. The system determines the (ξ, η, ζ) values corresponding to the glove's (x, y, z) coordinates using a table search and then refines them by inverse interpolation. First, it determines the nearest grid vertex to the given (x, y, z) glove coordinates through a simple two-step search of the vertex array. This gives a (ξ, η, ζ) value for that vertex. Then, it performs a first-order inverse interpolation using the (x, y, z) values of the neighboring (ξ, η, ζ) points to determine the (ξ, η, ζ) values of the glove position.

This technique fails on grids that are simultaneously highly curved and stretched. First, the search involves finding the nearest grid point to the seed position in physical space, and periodic grids may give false nearest points. Second, highly stretched grids cause the first-order interpolation to fail. We are presently implementing a more general method.

Once the seed point's computational space coordinates are known, a visualization tool uses these coordinates as initial conditions for integration, as described earlier. The system generates trajectories by integrating a system of three ordinary differential equations:

$$d\xi/dt = u(\xi, \eta, \zeta, t)$$

$$d\eta/dt = v(\xi, \eta, \zeta, t)$$

$$d\zeta/dt = w(\xi, \eta, \zeta, t)$$

where u, v, and w are the components of the velocity vector, in computational coordinates, at the point (ξ, η, ζ) and the time t. The system obtains values within a grid cell by trilinear interpolation of the values at the cell vertices.

In all cases, we use second-order accurate Runge-Kutta integration with a fixed step size. An integration is considered complete when a specified maximum number of steps has been computed, a position exceeds the grid boundaries, or the "end of time" is reached for nonperiodic flows. All integration is performed in the computational coordinate system.

After integration, the system transforms the computational coordinates of each visible point back to physical space and displays the points, possibly connected by lines. The whole procedure is fast enough to drag rakes of particle paths, streamlines, or streaklines through the field interactively, and it is inaccurate only near extreme velocity or metric discontinuities.

While the system performs these computations, it must have the vector field in physical memory. Since second-order Runge-Kutta integration is a predictor method, the vector-field data for both the current time step and the next time step are required. Because we want each successive time step to be computed and displayed at the rate of at least eight frames per second, the entire vector-field data set for all time is kept resident in physical memory. (We discuss the issue of accessing the data from disk at speeds sufficient for real-time visualization in a later section.)

Results

We used the implementation described here to visualize the 3D unsteady flow past a tapered cylinder computed by Jespersen and Levit.[5] The flow exhibits phenomena such as periodic vortex shedding, vortex tearing, and recirculation. This solution of the compressible Navier-Stokes equations consists of 800 successive time steps on a $64 \times 64 \times 32$ computational grid (131,072 grid points per time step). Each time step contains about 1.5 Mbytes of velocity data. Thus, the total size of the unsteady velocity field is over 1,000 Mbytes. As we said, to achieve a sufficient update rate for effective interaction, the entire data set must reside in physical memory. We truncate the tapered-cylinder data set in a variety of ways to fit the system's 256-Mbyte memory. One way is to use all time steps and truncate the spatial extent of the grid at each time step. We can select spatial subsets of the grid that exhibit interesting behavior. Truncating to a $42 \times 50 \times 10$ grid produces a data set with 800 time steps that is about 200 Mbytes in size and has sufficient spatial extent to exhibit interesting phenomena. Another choice is to truncate in time only. Displaying the entire grid for 130 time steps also produces a 200-Mbyte data set. The use of spatial truncation is a stopgap measure and must be done very carefully so as not to miss interesting flow phenomena. With this proviso, spatial truncation is one general way to load various data sets into resident memory.

We compared the flow phenomena observed with the virtual wind tunnel to those observed by Jespersen and Levit. We observed the same large-scale and many smaller scale features. Differences are presumably due to the use of the faster second-order Runge-Kutta integration methods, rather than more accurate but slower methods.

We considered subsampling as a way of fitting the data set into memory. We performed subsampling by a factor of 2 and compared the resulting flow phenomena with those seen by Jespersen and Levit. It was apparent that subsampling caused a severe distortion of the flow.

Numerical integration is distributed over the eight processors in the computer. In the case we describe here, when as many as 20,000 bubbles are present in the flow, the frame rate is about eight frames per second, which is barely sufficient for real-time interaction. We obtained this performance entirely in C with standard compiler optimizations. When numerical integration is halted, the frame rate increases.

The ability to define and move rakes of various types in the flow field lets the user quickly get an intuitive feel for the geometry of the velocity vector field. Particularly useful is the availability of rakes of different types. With a rake of bubblers, the user can identify the location of vortices or vortex tearing in space and time. For example, time can be frozen at the vortex-tearing event, freezing the bubbles that indicate its location. A new rake of streamlines can be added. When moved by hand, it gives a very nice method of exploring the geometry of the velocity vector field during the vortex-tearing event. Stepping time with well-placed rakes of streamlines inside the vortex as exhibited by the bubbler rake shows how that geometry evolves in time.

Performance issues

The study of interesting unsteady flows involves very large data sets. The full tapered-cylinder data set considered here is small, as these data go. Other data sets have larger grids, multiple grids, and more time steps. A typical engineering calculation might have two million grid points per time step and thousands of time steps. Thus, the requirement that the entire data set reside in physical memory is clearly an impediment to progress.

We might want to trade disk bandwidth for memory. One approach is to store only the solution for the current time step and replace it by reading new solutions from disk as virtual time

advances. Higher order integration methods that require two or more time steps simultaneously in memory still require only one new solution per time step. For this approach to succeed, the system must be able to read one 3D velocity field from disk for each frame of graphics displayed. For "slow-motion" effects, the required data rates are less, since several frames of flow visualization can be generated by interpolating between the solutions at two different time steps. However, in the most common case a new time step's worth of data needs to be read to generate a new frame of graphics.

Since the virtual wind tunnel uses a display sensitive to head position, the frame rate must not be allowed to slow down excessively. A good rule of thumb is to require at least 10 frames per second. Thus, the disk bandwidth required for applying this "out-of-core" approach for the full tapered-cylinder data set is 1.5 Mbytes/time step * 10 time steps/sec. = 15 Mbytes/sec.

The major computational component of the real-time visualization process in the virtual wind tunnel is numerical integration of streaklines, streamlines, and particle paths. This numerical integration consists largely of evaluating the integrands, which involves trilinear interpolations. Each second-order Runge-Kutta time-step advance for each particle involves about 200 floating-point operations (this includes transforming the particle's position back to physical coordinates for rendering). Again, keeping in mind the desired 10 frames per second and assuming that the Iris 380 VGX workstation is capable of 20 Mflops sustained, we can expect to process 10,000 particles per frame. Our current performance is somewhat better than this.

Using a large number of particles aids visualization, especially with bubblers, since the individual particles in a streakline usually spread too far apart to be connected by lines. Previous experience has shown us that 30,000 particles per frame may not be enough. A useful number is 100,000 per frame, but that requires 200 Mflops to support 10 frames per second. Thus, the integration of particle positions might profitably be distributed to a faster machine.

For a small model problem such as the tapered cylinder, a modern superworkstation with striped disks might attain the required performance. For larger problems, this approach requires a minisupercomputer or better. An example engineering problem that we will address is the visualization of a data set of a hovering Harrier jump jet. This data set involves hundreds of time steps with 21 grids at each time step for a total of about a million points per time step. The data size is 36 Mbytes per time step. Visualization of this data set in the virtual wind tunnel is our grand challenge for 1992.

Future directions

During 1992 we are experimenting with a variety of solutions to this problem. Our primary direction is to distribute the computation of the streaklines, streamlines, and so on to a supercomputer over high-speed networks. The distribution over the network has been implemented via Gerald-Yamasaki's distributed library routines.[12] This library wraps network communications in simple function calls, greatly facilitating the design of a distributed application. The graphics workstation will still be responsible for the rendering and user interface, as in the current virtual wind tunnel. As a user moves a rake, the system will send the positions of the user's hand and fingers to the supercomputer. The supercomputer will compute the evolution of the particles in the currently working tools and send these positions back to the workstation for rendering. To be successful, all this must happen at a rate of about 10 Hz. This architecture has a variety of advantages, including higher computational speed so more particles can be used for visualization, larger physical memory, and higher disk bandwidth.

While distribution of the virtual wind tunnel will give some advantages, preliminary estimates indicate that both disk bandwidth and network bandwidth will be bottlenecks.

While distribution of the virtual wind tunnel will give some advantages, preliminary estimates indicate that both disk bandwidth and network bandwidth will be bottlenecks. Neither the Cray nor Convex supercomputer seems to have the disk bandwidth required to work with 36-Mbyte data sets in real time. They do, however, have the disk bandwidth to look at smaller data sets (approximately 1 to 2 Mbytes per time step) for a very large number of time steps. Data sets of this type, such as the tapered cylinder, exist. The network bottleneck is somewhat less severe. The Ultranet bandwidth lets the current workstation read at 10 Mbytes per second. This rate is sufficient for tens of thousands of particles, but additional bandwidth is desirable.

Another architecture we will develop is inspired by the fact that supercomputers have potentially large physical memory sizes. The Convex system at NASA Ames has a physical memory of 1 Gbyte. Thus, by distributing to the Convex, we can have 27 or so time steps of the Harrier data set resident in memory. This may be enough to look at phenomena of interest.

Both architectures are worth exploring, even if their bottlenecks prevent a useful distributed virtual wind tunnel. The bottlenecks are technological and will diminish with advances in the hardware. Our exploration of these architectures will pave the way for the time when the hardware is capable of the performance we require.

High-speed disks (up to 100 Mbytes per second bandwidth) are being developed and might work with next-generation

workstations. They might allow us to use a virtual wind tunnel linked to a disk-based stand-alone workstation to look at medium-size interesting data sets.

With the new two-color-channel color boom described earlier, we require a workstation able to output two full-color graphics signals. This will be performed by the new Silicon Graphics Skywriter workstation, which has two independent VGX graphics channels, controlled by four CPUs. While this system has somewhat lower computational power than the existing system, it has greater graphics capacity and will be able to drive the color boom. In the distributed virtual wind tunnel, where the computationally intensive tasks are distributed to a high-performance system, this lack of computational power at the workstation should not be a problem.

Once the distributed virtual wind tunnel is working, it should be easy to extend it so that several workstations with boom and glove can display the same data set resident on the supercomputer. This will allow a shared virtual wind tunnel in which potentially distant researchers can interact with and visualize the same data set. We will turn to this work in mid-1992.

Finally, there is the question of determining the extent to which the virtual wind tunnel is actually useful to researchers. Some indication of this is that the virtual wind tunnel hardware has been duplicated at the National Center for Supercomputing Applications at the University of Illinois in Urbana and at the Army Corps of Engineers Waterways Experimental Station in Vicksburg, Miss. In collaboration with us, both sites will use the concepts described here to look at various types of flows, specifically, cosmic magnetohydrodynamic jets at NCSA and water channel flows at WES. Thus, there are the beginnings of a user community.

The existence of a user community does not demonstrate the extent to which the virtual wind tunnel is actually useful, however. This will require formal studies and experiments. We feel that the virtual wind tunnel interface and capabilities should be further enhanced before these studies are carried out. We expect to be ready for these studies at the end of 1992, and will carry them out in collaboration with the human factors division at NASA Ames.

Virtual environments are a tantalizing new medium that may become important to scientists and engineers. Although they are currently rather lackluster in terms of performance and image quality, the situation is reminiscent of that in scientific visualization, say, 10 years ago, when only slow, relatively low-quality output devices were available. It was not that long ago when scientific visualization on computers was done by clever overprinting with line printers, or, if one was really lucky, with five different colored pens on a CalComp plotter. Virtual environments can be viewed as extensions of trends already in existence for years: faster 3D rendering, bigger screens, the "desktop" metaphor, and much else. The "sense of presence" produced by even today's virtual environments certainly has implications for entertainment and possibly for education. We believe that virtual environments also have great application in the field of scientific visualization in general. Virtual environments is still a young field, and the virtual wind tunnel is a pioneering effort in the application of this technology to understanding complex phenomena. □

Acknowledgment

This work was supported under government contract NAS 2-12961.

References

1. M.H. Smith et al., "Analysis and Visualization of Complex Unsteady Three-Dimensional Flows," Paper AIAA-89-0139, *Ann. Aerospace Sciences Meeting*, Am. Inst. of Aeronautics and Astronautics, New York, 1989.
2. R. Haimes and M. Giles, "VISUAL3: Interactive Unsteady Unstructured 3D Visualization," Paper AIAA-91-0794, *Ann. Aerospace Sciences Meeting*, Am. Inst. of Aeronautics and Astronautics, New York, 1991.
3. S. Shirayama, "Visualization of Vector Fields in Flow Analysis I," Paper AIAA-91-0801, *Ann. Aerospace Sciences Meeting*, Am. Inst. of Aeronautics and Astronautics, New York, 1991.
4. W.J. Yang, ed., *Handbook of Flow Visualization*, Hemisphere Publishing, New York, 1989.
5. D. Jespersen and C. Levit, "Numerical Simulation of Flow Past a Tapered Cylinder," Paper AIAA-91-0751, *Ann. Aerospace Sciences Meeting*, Am. Inst. of Aeronautics and Astronautics, New York, 1991.
6. J.P. Crowder, "Tufts," *Handbook of Flow Visualization*, Hemisphere Publishing, New York, pp. 125-175.
7. P.K. Kundu, *Fluid Mechanics*, Academic Press, San Diego, Calif., 1990, pp. 51-53.
8. A. Globus, T. Lasinski, and C. Levit, "A Tool for Visualizing the Topology of Vector Fields," *Proc. IEEE Visualization 91*, CS Press, Los Alamitos, Calif., 1991, pp. 17-24.
9. L.A. Yates and G.T. Chapman, "Streamlines, Vorticity Lines, and Vortices," Paper AIAA-91-0731, *Ann. Aerospace Sciences Meeting*, Am. Inst. of Aeronautics and Astronautics, New York, 1991.
10. I.E. McDowall et al., "Implementation and Integration of a Counterbalance CRT-Based Stereoscopic Display for Interactive Viewpoint Control in Virtual Environment Application," *Proc. SPIE Conf. Stereoscopic Displays and Applications*, SPIE, Bellingham, Wash., 1990.
11. S. Fisher et al., "Virtual Environment Interface Workstations," *Proc. Human Factors Soc. Ann. Meeting*, 1988.
12. M. Gerald-Yamasaki, "Distributed Library," NAS Applied Research Tech. Report RNR-90-008, 1990.

Steve Bryson does research in the use of advanced interfaces and data display techniques for the visualization of numerically simulated fluid flows at NASA Ames Research Center. This work primarily involves virtual environment techniques, which Bryson has ben actively developing for more than eight years. He is also pursuing visualization of other physical phenomena.

Creon Levit received his BS in computer science from Washington University in St. Louis in 1982. He joined NASA Ames Research Center later that year. For the last several years, he has been pursuing research in computational aerodynamics on massively parallel computers and in numerical flow visualization using computer graphics and virtual environments.

The authors can be reached at the Applied Research Office, Numerical Aerodynamics Simulation Division, NASA Ames Research Center, MS T045-1, Moffett Field, CA 94035